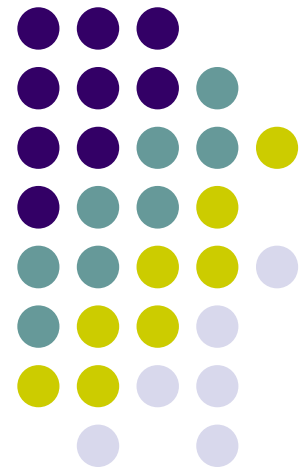


Access Control for Federation of Emulab-based Network Testbeds

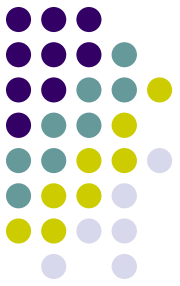
Ted Faber, John Wroclawski

28 July 2008

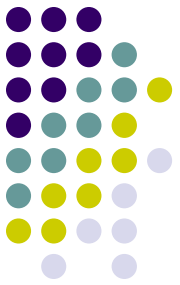
faber@isi.edu, jtw@isi.edu



Outline



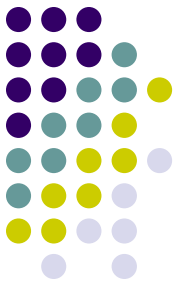
- Testbed Federation
- DETER federation model and architecture
- Access control in the DETER architecture
- Access control implementation: fedd



Federation of Network Testbeds

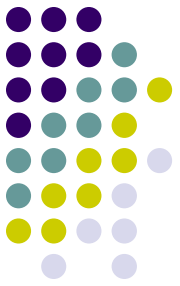
- Federation: Independent testbeds working together
 - Testbeds maintain local control of shared resources
- Inherent tension:
 - Experimenters' needs
 - Testbeds' constraints
- Inherent power:
 - More resources
 - More kinds of resources
 - More environments

Federation's Power: not just more resources



- Federation enables
 - Combining testbed resources
 - Hardware
 - Tools
 - Combining testbed environments
 - Operational features – security properties
 - Combining testbed communities
 - Collaborative experiments
 - Competitive exploration
- Individual testbed properties come from testbed independence

The Critical Capabilities



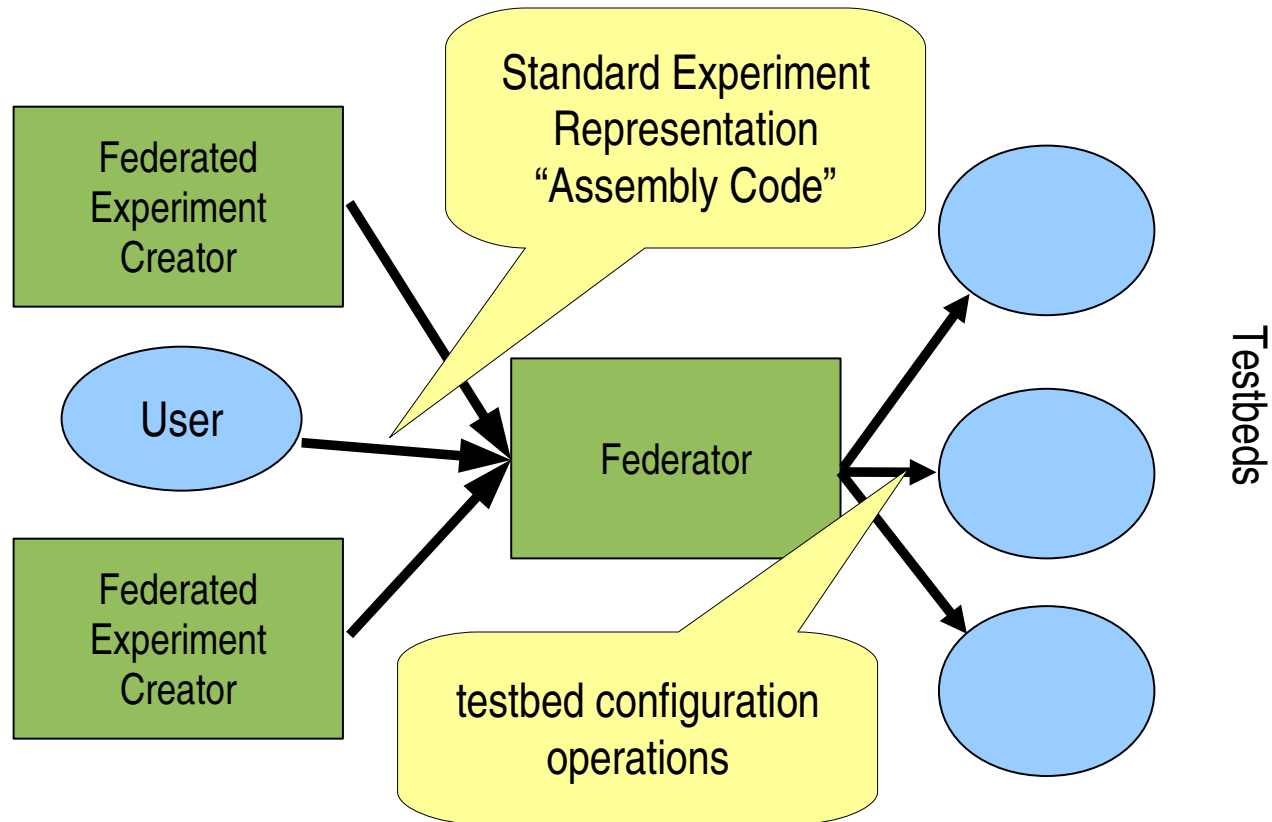
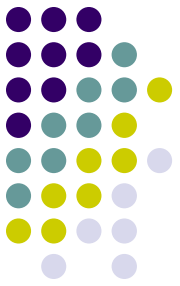
- Helping experimenters federate
 - Fitting experiments into federation
 - Fitting federation into experiments
- **Letting testbeds control their resources**
 - **Sharing information about experimenters**
 - **Controlling access to resources**
- Supporting federated experiments
 - Powerful tools
 - Usable environments

DETER Federation Model



- Experiments are created from federated resources
 - Minimal prior agreements between testbeds
 - Specifically humans are not involved per-experiment
 - Testbeds have emulab interfaces
- Federated experiments are conducted in emulab-like environment
 - Ssh logins
 - Shared filesystems
 - Emulab event system
- Model Goal: create a familiar environment appropriate for the experiment from independent testbeds

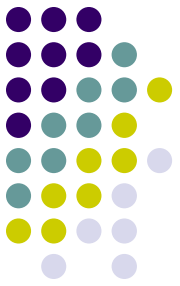
DETER Federation Architecture



- Flow:

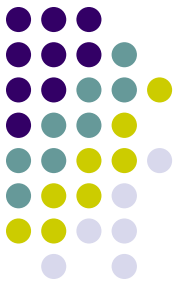
- Experimenter: think, design, experiment
- Federator: decompose, embed, connect
- Testbed: authenticate, allow, allocate

Making an Experiment



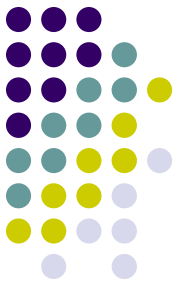
- Experimenter has hypothesis
- Turning a hypothesis to an experiment:
 - Selecting appropriate topology
 - Positioning experimental elements in it
 - Mapping those to physical elements
- Mapping complicated by federation
 - Resource discovery
 - **Access requirements**
 - Experimental effects
- Each automated phase improves access

Experiment Creation: Mapping an Experiment



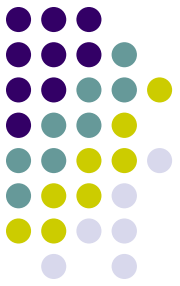
- User assisted decomposition:
 - User understands federation and experiment
 - Suggests possible splittings to tools
- Automated decomposition:
 - Tools detect areas of homogeneity and split
 - Domain-specific definitions of homogeneity
- Experiment generation:
 - Loose description in domain specific-language
 - Generate and split experiment from it
 - Pipeline into decomposition

Access Control: Putting global users into local scope



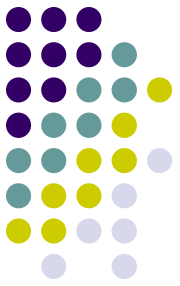
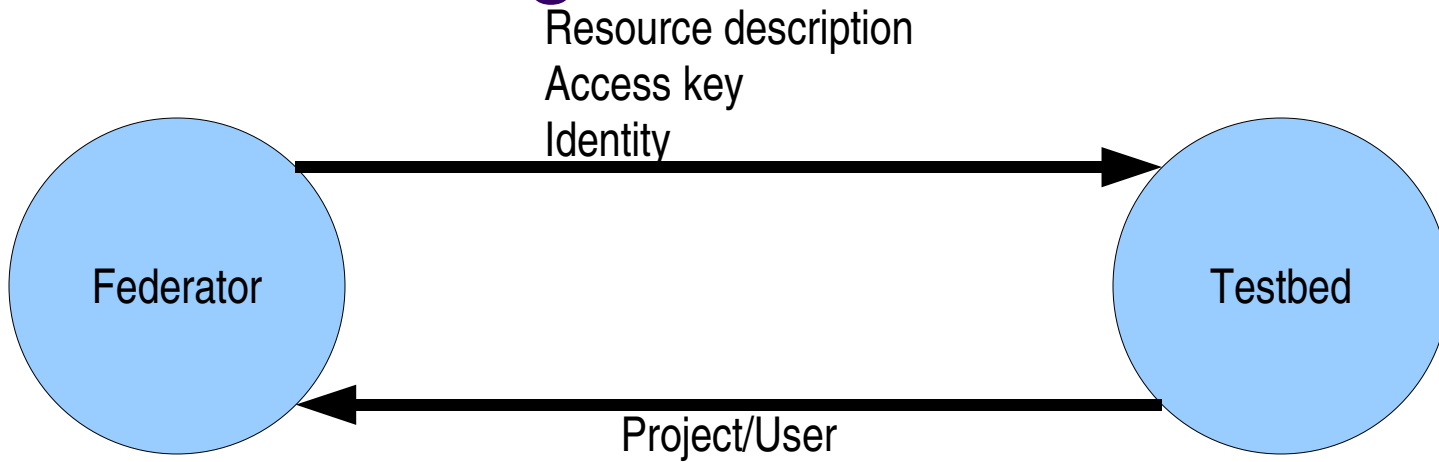
- Resources are controlled locally
- Federated users must be integrated into local:
 - Representations
 - Policies
 - Accounting
- After access control exchange:
 - Federated experimenter can authenticate to testbed
 - Testbed knows experimenter's rights in local terms
- Analogous to getting an account on a system

DETER Access Control Architecture



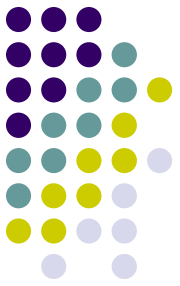
- Based on single-emulab structures:
 - Projects control resource access
 - User's project membership determines access
- DETER federation architecture:
 - Users, projects, testbeds have global names
 - Federants permit accesses (create accounts) based on:
 - Proof of name
 - Attested facts (evaluated wrt name)
 - Local information bound to name
 - Federants assign federated experiments to local projects

Access Exchange



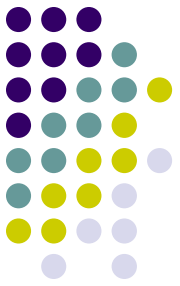
- Access key: SSH public key
- Project/User has rights to requested resources
- Project/User configured to be accessible by Access key

Global Entities: Users, Projects, Testbeds



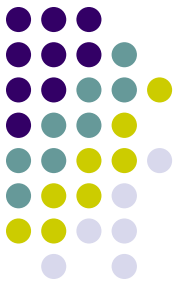
- Basis of a three-level access control system
- Each has different attestation abilities
 - Testbeds about itself, its projects, and its users
 - Projects about itself and its users
 - Users about themselves
- A user making a request can
 - Ask directly
 - Ask its project to ask for it (use a project name)
 - Ask its testbed to ask for it (use a testbed name)
- Attestation improves scalability

Self-certifying Names



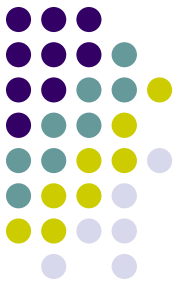
- Properties:
 - Holder can prove identity interactively
 - Others agree on holder's identity
- Many implementations (self-signed PGP key)
- Provide a global decentralized naming system
 - Scale
- Federants access info tied to names
 - Flexibility

Fedd: Platform for Access Control



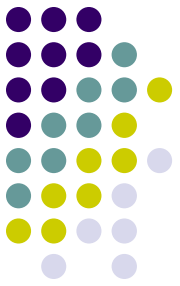
- Implementation of the DETER access control arch.
- Open interfaces: WSDL/SOAP descriptions
 - We have 2 interoperable implementations
- Prototype implementation of global identifiers for federation
- Flexible access control descriptions in emulab terms
 - Dynamic project allocation

Federation ID Implementation



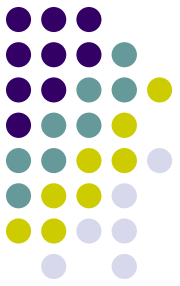
- ID is hash of a public key
 - Hash obscures key format, length for versatility
- Passed between Federator and testbed as X.509 certificates
 - Self-signed certificate – self-certifying name
 - NB: name as hash provides more assurance
 - Can use certification chain as introducer
 - Binding information to ID through trusted chain
- Direct TLS/SSL implementation

Access Control Configuration



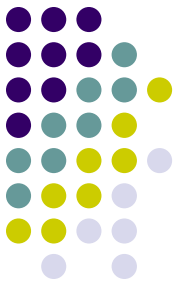
- Map from (testbed, project, user) to local (project, user)
- Abstractly:
 - Maps from global namespace -> local namespace
 - Realizes access control decisions as local rights
- Concretely
 - Provides access using familiar access controls
 - Maintains local control over resources
- An access control request may result in local project creation

Expressing Access Decisions



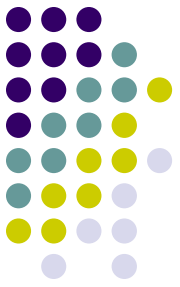
- Fedd access rules
 - Known user to static project:
 - (deter, DeterTest, faber) -> (fDeter, faber)
 - Any user from testbed to static project, new user:
 - (deter, <any>, <any>) -> (fDeter, <dynamic>)
 - Project annotated with access:
 - (deter, <any>, <any>) -> (fDeter:special_type, <dynamic>)
 - Anonymous users:
 - (<none>, <none>, <any>) -> (lockdown, anon)

Dynamic Project Allocation



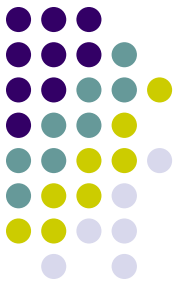
- Fedd can create projects for federated experiments
- Dynamic projects:
 - Precise allocation of rights to federated experiment
 - Clear accounting of federated actions
 - More system resources & startup time
- Static projects:
 - Amortize system costs for federation support
 - Naturally support discrete access classes
- No human intervention required

Future Work: Realizing more of the Architecture



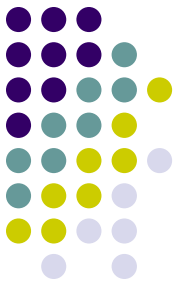
- Tools for experiment creation
 - Network topology-based embedders
 - Domain specific embedders
 - Experiment Creation/Topology Generation tools
- Resource advertisements
- Experiment Environment
 - More scalable file system
 - More natural model for experimentation in the large

Conclusions



- Presented model for experiments that supports
 - Larger experiments
 - New kinds of experiments
- DETER federation architecture
 - Framework for creating such experiments
- Particular piece of that architecture: Access Control
 - Abstract framing
 - Current implementation

Attestation and Amplification



- Projects and Testbeds
 - May have more privileges than users
 - May be more constrained than users
 - “DETER testbed may use any hardware, but only 10 total machines”
- Users negotiate with project & testbed agents
 - Agent will make request with its credentials
 - Agent may impose additional constraints